State Complexity of Deletion

Yo-Sub Han[1] **Sang-Ki Ko**[1] Kai Salomaa[2]

[1]Department of Computer Science
Yonsei University

[2]School of Computing
Queen's University

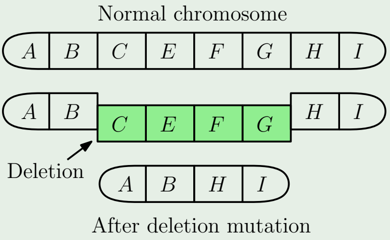
DLT 2014, Ekaterinburg

Overview

Deletion

- Deletion is one of the basic operations in formal language theory.
- The deletion of a string v from a string u consists of erasing a **contiguous substring** v from u.

Example (Deletion mutation in genes)



Deletion along Trajectories

Example

Let $x = aabbcc$, $y = abc$ and $t = (id)^3$. Then, we have that $x \rightsquigarrow_t y = abc$. If $t = i^3d^3$, then $x \rightsquigarrow_t y = \emptyset$.

$$
\begin{array}{rcllllll}
x & = & a & a & b & b & c & c \\
t & = & i & d & i & d & i & d \\
y & = & & a & & b & & c \\
x \rightsquigarrow_t y & = & & & & & &
\end{array}
$$

Deletion along Trajectories

Example

Let $x = aabbcc$, $y = abc$ and $t = (id)^3$. Then, we have that $x \rightsquigarrow_t y = abc$. If $t = i^3d^3$, then $x \rightsquigarrow_t y = \emptyset$.

$$\begin{array}{rcllllll} x & = & a & a & b & b & c & c \\ t & = & i & d & i & d & i & d \\ y & = & & a & & b & & c \\ x \rightsquigarrow_t y & = & & & & & & \end{array}$$

Deletion along Trajectories

Example

Let $x = aabbcc$, $y = abc$ and $t = (id)^3$. Then, we have that $x \rightsquigarrow_t y = abc$. If $t = i^3d^3$, then $x \rightsquigarrow_t y = \emptyset$.

$$\begin{array}{rcccccccc} x & = & a & a & b & b & c & c \\ t & = & i & d & i & d & i & d \\ y & = & & a & & b & & c \\ x \rightsquigarrow_t y & = & a & & b & & c & \end{array}$$

Deletion along Trajectories

Example

Let $x = aabbcc$, $y = abc$ and $t = (id)^3$. Then, we have that $x \rightsquigarrow_t y = abc$. If $t = i^3d^3$, then $x \rightsquigarrow_t y = \emptyset$.

$$\begin{array}{rcllllll} x & = & a & a & b & b & c & c \\ t & = & i & i & i & d & d & d \\ y & = & & & & a & b & c \\ x \rightsquigarrow_t y & = & & & & & & \end{array}$$

Deletion along Trajectories

Example

Let $x = aabbcc$, $y = abc$ and $t = (id)^3$. Then, we have that $x \rightsquigarrow_t y = abc$. If $t = i^3 d^3$, then $x \rightsquigarrow_t y = \emptyset$.

$$
\begin{array}{rcllllll}
x & = & a & a & b & b & c & c \\
t & = & i & i & i & d & d & d \\
y & = & & & & a & b & c \\
x \rightsquigarrow_t y & = & & & & & &
\end{array}
$$

Deletion along Trajectories

Definition

Let $T \subseteq \{i, d\}^*$. Then,

$$x \rightsquigarrow_T y = \bigcup_{t \in T} x \rightsquigarrow_t y.$$

We can extend to the languages!

Definition

Let $L_1, L_2 \subseteq \Sigma^*$ and $T \subseteq \{i, d\}^*$. Then

$$L_1 \rightsquigarrow_T L_2 = \bigcup_{x \in L_1, y \in L_2} x \rightsquigarrow_T y.$$

Deletion along Trajectories

Definition

Let $T \subseteq \{i, d\}^*$. Then,

$$x \rightsquigarrow_T y = \bigcup_{t \in T} x \rightsquigarrow_t y.$$

We can extend to the languages!

Definition

Let $L_1, L_2 \subseteq \Sigma^*$ and $T \subseteq \{i, d\}^*$. Then

$$L_1 \rightsquigarrow_T L_2 = \bigcup_{x \in L_1, y \in L_2} x \rightsquigarrow_T y.$$

Deletion and Left/Right Quotient

- Deletion is the **simplest** and **most natural** generalization of the left/right quotient.
- Left quotient is deletion along a set of trajectories d^*i^*.
  $(L_2 \setminus L_1 = L_1 \rightsquigarrow_{d^*i^*} L_2)$
- Right quotient is deletion along a set of trajectories i^*d^*.
  $(L_1 / L_2 = L_1 \rightsquigarrow_{i^*d^*} L_2)$
- Here we consider the deletion along a set of trajectories $i^*d^*i^*$.

State Complexity

Definition

The **state complexity** of L, $\mathrm{sc}(L)$, is the size of the minimal complete DFA recognizing L.

Definition

The **incomplete state complexity** of L, $\mathrm{isc}(L)$, is the size of the minimal incomplete DFA recognizing L.

For each regular language L either

$$\mathrm{sc}(L) = \mathrm{isc}(L) + 1$$

or

$$\mathrm{sc}(L) = \mathrm{isc}(L).$$

State Complexity of Left/Right Quotient

Known results (S. Yu, 1997)

It is known that for L_1 recognized by a DFA with n states and an arbitrary language L_2, the worst case state complexity of

- the left-quotient $L_2 \setminus L_1$ is $2^n - 1$ and
- the state complexity of the right-quotient L_1/L_2 is n.

Problem

What is the precise state complexity of deletion (along a set of trajectories $i^*d^*i^*$)?

State Complexity of Left/Right Quotient

Known results (S. Yu, 1997)

It is known that for L_1 recognized by a DFA with n states and an arbitrary language L_2, the worst case state complexity of

- the left-quotient $L_2 \setminus L_1$ is $2^n - 1$ and
- the state complexity of the right-quotient L_1/L_2 is n.

Problem

What is the precise state complexity of deletion (along a set of trajectories $i^*d^*i^*$)?

Deletion Preserves Regularity

Theorem (L. Kari, 1994)

It is well known that $L_1 \rightsquigarrow L_2$ *is always regular for a regular language* L_1 *and an arbitrary language* L_2*.*

Corollary (L. Kari, 1994)

The language $L_1 \rightsquigarrow L_2$ *can be effectively constructed if* L_1 *is a regular language and* L_2 *is a regular or context-free language.*

Note

The proof of the theorem yields an upper bound 2^{2n} which works for an arbitrary language L_2 and is not **effective**.

Deletion Preserves Regularity

Theorem (L. Kari, 1994)

It is well known that $L_1 \rightsquigarrow L_2$ is always regular for a regular language L_1 and an arbitrary language L_2.

Corollary (L. Kari, 1994)

The language $L_1 \rightsquigarrow L_2$ can be effectively constructed if L_1 is a regular language and L_2 is a regular or context-free language.

Note

The proof of the theorem yields an upper bound 2^{2n} which works for an arbitrary language L_2 and is not **effective**.

Upper Bound for Complete DFAs

Lemma

Consider $L_1, L_2 \subseteq \Sigma^*$ *where* L_1 *is recognized by a complete DFA with* n *states. Then*

$$\mathrm{sc}(L_1 \rightsquigarrow L_2) \leq n \cdot 2^{n-1}.$$

Proof for Upper Bound

Complete DFAs

Proof.

Let $A = (Q, \Sigma, \delta, q_0, F_A)$ be a complete DFA for L_1 where $|Q| = n$.

To recognize the language $L_1 \rightsquigarrow L_2$ we define a DFA

$$B = (P, \Sigma, \gamma, p_0, F_B),$$

where

- $P = \{(r, R) \mid r \in Q, R \subseteq Q, \delta(\mathbf{r}, \mathbf{L_2}) \subseteq \mathbf{R}\}$, $(|P| = n \cdot 2^n)$
- $p_0 = (q_0, \delta(\mathbf{q_0}, \mathbf{L_2}))$ and
- $F_B = \{(r, R) \mid r \in Q, R \subseteq Q, \delta(\mathbf{r}, \mathbf{L_2}) \subseteq \mathbf{R}$ and $\mathbf{R} \cap \mathbf{F_A} \neq \emptyset\}$.

Proof for Upper Bound

Complete DFAs

Proof.

Let $A = (Q, \Sigma, \delta, q_0, F_A)$ be a complete DFA for L_1 where $|Q| = n$.

To recognize the language $L_1 \rightsquigarrow L_2$ we define a DFA

$$B = (P, \Sigma, \gamma, p_0, F_B),$$

where

- $P = \{(r, R) \mid r \in Q, R \subseteq Q, \delta(\mathbf{r}, \mathbf{L_2}) \subseteq \mathbf{R}\}$,

  $(|P| = n \cdot 2^n)$
- $p_0 = (q_0, \delta(\mathbf{q_0}, \mathbf{L_2}))$ and
- $F_B = \{(r, R) \mid r \in Q, R \subseteq Q, \delta(\mathbf{r}, \mathbf{L_2}) \subseteq \mathbf{R}$ and $\mathbf{R} \cap \mathbf{F_A} \neq \emptyset\}$.

Proof for Upper Bound

Complete DFAs

Proof.

Let $A = (Q, \Sigma, \delta, q_0, F_A)$ be a complete DFA for L_1 where $|Q| = n$.

To recognize the language $L_1 \rightsquigarrow L_2$ we define a DFA

$$B = (P, \Sigma, \gamma, p_0, F_B),$$

where

- $P = \{(r, R) \mid r \in Q, R \subseteq Q, \delta(\mathbf{r}, \mathbf{L_2}) \subseteq \mathbf{R}\}$,

  $(|P| = n \cdot 2^n)$

- $p_0 = (q_0, \delta(\mathbf{q_0}, \mathbf{L_2}))$ and
- $F_B = \{(r, R) \mid r \in Q, R \subseteq Q, \delta(\mathbf{r}, \mathbf{L_2}) \subseteq \mathbf{R}$ and $\mathbf{R} \cap \mathbf{F_A} \neq \emptyset\}$.

Proof for Upper Bound

Complete DFAs

Proof.

Let $A = (Q, \Sigma, \delta, q_0, F_A)$ be a complete DFA for L_1 where $|Q| = n$.

To recognize the language $L_1 \rightsquigarrow L_2$ we define a DFA

$$B = (P, \Sigma, \gamma, p_0, F_B),$$

where

- $P = \{(r, R) \mid r \in Q, R \subseteq Q, \delta(\mathbf{r}, \mathbf{L_2}) \subseteq \mathbf{R}\}$,

  $(|P| = n \cdot 2^n)$
- $p_0 = (q_0, \delta(\mathbf{q_0}, \mathbf{L_2}))$ and
- $F_B = \{(r, R) \mid r \in Q, R \subseteq Q, \delta(\mathbf{r}, \mathbf{L_2}) \subseteq \mathbf{R}$ and $\mathbf{R} \cap \mathbf{F_A} \neq \emptyset\}$.

Proof for Upper Bound

Complete DFAs

Proof.

It remains to define the transitions of γ. For $(r, R) \in P$ and $b \in \Sigma$ we set

$$\gamma((r, R), b) = (\delta(r, b), \quad \delta(R, b) \cup \delta(\delta(\mathbf{r}, \mathbf{b}), \mathbf{L_2})).$$

$$\underbrace{abac\,\overbrace{ababacca}^{u \in L_2}\,bac}_{w \in L_1}$$

□

Proof for Upper Bound
Complete DFAs

Proof.

It remains to define the transitions of γ. For $(r, R) \in P$ and $b \in \Sigma$ we set

$$\gamma((r, R), b) = (\delta(r, b),\ \ \delta(R, b) \cup \delta(\delta(\mathbf{r}, \mathbf{b}), \mathbf{L_2})).$$

$$\underbrace{abac\,\overbrace{ababacca}^{u \in L_2}\,bac}_{w \in L_1}$$

□

Proof for Upper Bound

Complete DFAs

Proof.

It remains to define the transitions of γ. For $(r, R) \in P$ and $b \in \Sigma$ we set

$$\gamma((r, R), b) = (\delta(r, b), \quad \delta(R, b) \cup \delta(\delta(\mathbf{r}, \mathbf{b}), \mathbf{L_2})).$$

$$\underbrace{aba c \overbrace{abab acca}^{u \in L_2} bac}_{w \in L_1}$$

□

Proof for Upper Bound
Complete DFAs

Proof.

It remains to define the transitions of γ. For $(r, R) \in P$ and $b \in \Sigma$ we set

$$\gamma((r,R),b) = (\delta(r,b),\quad \delta(R,b) \cup \delta(\delta(\mathbf{r},\mathbf{b}),\mathbf{L_2})).$$

$$\underbrace{abac\,\overbrace{ababacca}^{u\in L_2}\,bac}_{w\in L_1}$$

$\square$

Proof for Upper Bound
Complete DFAs

Proof.

It remains to define the transitions of γ. For $(r, R) \in P$ and $b \in \Sigma$ we set

$$\gamma((r, R), b) = (\delta(r, b), \quad \delta(R, b) \cup \delta(\delta(\mathbf{r}, \mathbf{b}), \mathbf{L_2})).$$

$$\underbrace{abac\,\overbrace{abababacca}^{u \in L_2}\,bac}_{w \in L_1}$$

□

Proof for Upper Bound

Complete DFAs

Proof.

It remains to define the transitions of γ. For $(r, R) \in P$ and $b \in \Sigma$ we set

$$\gamma((r, R), b) = (\delta(r, b), \quad \delta(R, b) \cup \delta(\delta(\mathbf{r}, \mathbf{b}), \mathbf{L_2})).$$

$$\underbrace{abac\,\overbrace{ababacca}^{u \in L_2}\,bac}_{w \in L_1}$$

□

Proof for Upper Bound

Complete DFAs

Proof.

It remains to define the transitions of γ. For $(r, R) \in P$ and $b \in \Sigma$ we set

$$\gamma((r, R), b) = (\delta(r, b), \quad \delta(R, b) \cup \delta(\delta(\mathbf{r}, \mathbf{b}), \mathbf{L_2})).$$

$$\underbrace{abac\,\overbrace{abababacca}^{u \in L_2}\,bac}_{w \in L_1}$$

□

Proof for Upper Bound
Complete DFAs

Proof.

It remains to define the transitions of γ. For $(r, R) \in P$ and $b \in \Sigma$ we set

$$\gamma((r, R), b) = (\delta(r, b),\ \ \delta(R, b) \cup \delta(\delta(\mathbf{r}, \mathbf{b}), \mathbf{L_2})).$$

- Since $L_2 \neq \emptyset$, for each $r \in Q$, we have $|\delta(r, L_2)| \geq 1$.
- So, there exist at most 2^{n-1} sets R such that (r, R) is a state of B.
- As a result, we have an upper bound $n \cdot 2^{n-1}$.

□

Upper Bound for Incomplete DFAs

Lemma

Let $L_1, L_2 \subseteq \Sigma^*$ *where* L_1 *is recognized by an incomplete DFA* A *with* n *states. Then*

$$\text{isc}(L_1 \rightsquigarrow L_2) \leq (n+1) \cdot 2^n - (2^{n-1} + 2).$$

Proof for Upper Bound

Incomplete DFAs

Proof.

Let $A = (Q, \Sigma, \delta, q_0, F_A)$ be an incomplete DFA for L_1, $|Q| = n$.

We define the **completion** of δ as a function

$$\delta' : (Q \cup \{\text{dead}\}) \times \Sigma \to Q \cup \{\text{dead}\}$$

by setting for $r \in Q \cup \{\text{dead}\}$ and $b \in \Sigma$,

$$\delta'(r, b) = \begin{cases} \delta(r, b), & \text{if } r \in Q \text{ and } \delta(r, b) \text{ is defined;} \\ \text{dead}, & \text{otherwise.} \end{cases}$$

Proof for Upper Bound

Incomplete DFAs

Proof.

Let $A = (Q, \Sigma, \delta, q_0, F_A)$ be an incomplete DFA for L_1, $|Q| = n$.

We define the **completion** of δ as a function

$$\delta' : (Q \cup \{\text{dead}\}) \times \Sigma \to Q \cup \{\text{dead}\}$$

by setting for $r \in Q \cup \{\text{dead}\}$ and $b \in \Sigma$,

$$\delta'(r, b) = \begin{cases} \delta(r, b), & \text{if } r \in Q \text{ and } \delta(r, b) \text{ is defined;} \\ \text{dead}, & \text{otherwise.} \end{cases}$$

Proof for Upper Bound

Incomplete DFAs

Proof.

Define a DFA

$$B = (P, \Sigma, \gamma, p_0, F_B),$$

where

- $P = (Q \cup \{\text{dead}\}) \times 2^Q - \{(\text{dead}, \emptyset), (\text{dead}, Q)\}$,
  $(|P| = (n+1) \cdot 2^n - 2)$
- $p_0 = (q_0, \delta(q_0, L_2))$ and
  (same as complete case)
- $F_B = \{(r, R) \mid r \in Q \cup \{\text{dead}\}, R \subseteq Q$ and $R \cap F_A \neq \emptyset\}$.
  (same as complete case except $\{\text{dead}\}$)

Proof for Upper Bound

Incomplete DFAs

Proof.

Define a DFA

$$B = (P, \Sigma, \gamma, p_0, F_B),$$

where

- $P = (Q \cup \{\text{dead}\}) \times 2^Q - \{(\text{dead}, \emptyset), (\text{dead}, Q)\}$,
  $(|P| = (n+1) \cdot 2^n - 2)$
- $p_0 = (q_0, \delta(q_0, L_2))$ and
  (same as complete case)
- $F_B = \{(r, R) \mid r \in Q \cup \{\text{dead}\}, R \subseteq Q$ and $R \cap F_A \neq \emptyset\}$.
  (same as complete case except $\{\text{dead}\}$)

Proof for Upper Bound

Incomplete DFAs

Proof.

Define a DFA

$$B = (P, \Sigma, \gamma, p_0, F_B),$$

where

- $P = (Q \cup \{\text{dead}\}) \times 2^Q - \{(\text{dead}, \emptyset), (\text{dead}, Q)\}$,
  $(|P| = (n+1) \cdot 2^n - 2)$
- $p_0 = (q_0, \delta(q_0, L_2))$ and
  (same as complete case)
- $F_B = \{(r, R) \mid r \in Q \cup \{\text{dead}\}, R \subseteq Q$ and $R \cap F_A \neq \emptyset\}$.
  (same as complete case except $\{\text{dead}\}$)

Proof for Upper Bound

Incomplete DFAs

Proof.

The transitions of γ are defined by setting, for $(r, R) \in P$ and $b \in \Sigma$, we define $\gamma((r, R), b)$ to be

- $(\delta'(r, b),\ \ \delta(R, b) \cup \delta(\delta(r, b), L_2))$, if $r \in Q$ and $(\delta'(r, b) \neq \mathrm{dead}$ or $\delta(R, b) \cup \delta(\delta(r, b), L_2) \neq \emptyset)$;
- $(\mathrm{dead}, \delta(R, b))$, if $r = \mathrm{dead}$ and $\delta(R, b) \neq \emptyset$;
- undefined, otherwise.

We have 2^{n-1} unreachable states since for some $q_1 \in Q$ and $w_1 \in L_2$, $\delta(q_1, w_1)$ **must** be defined. □

Proof for Upper Bound

Incomplete DFAs

Proof.

The transitions of γ are defined by setting, for $(r, R) \in P$ and $b \in \Sigma$, we define $\gamma((r, R), b)$ to be

- $(\delta'(r, b),\ \ \delta(R, b) \cup \delta(\delta(r, b), L_2))$, if $r \in Q$ and $(\delta'(r, b) \neq \text{dead}$ or $\delta(R, b) \cup \delta(\delta(r, b), L_2) \neq \emptyset)$;
- $(\text{dead}, \delta(R, b))$, if $r = \text{dead}$ and $\delta(R, b) \neq \emptyset$;
- undefined, otherwise.

We have 2^{n-1} unreachable states since for some $q_1 \in Q$ and $w_1 \in L_2$, $\delta(q_1, w_1)$ **must** be defined. □

Proof for Upper Bound

Incomplete DFAs

Proof.

The transitions of γ are defined by setting, for $(r, R) \in P$ and $b \in \Sigma$, we define $\gamma((r, R), b)$ to be

- $(\delta'(r, b),\ \ \delta(R, b) \cup \delta(\delta(r, b), L_2))$, if $r \in Q$ and $(\delta'(r, b) \neq \text{dead}$ or $\delta(R, b) \cup \delta(\delta(r, b), L_2) \neq \emptyset)$;
- $(\text{dead}, \delta(R, b))$, if $r = \text{dead}$ and $\delta(R, b) \neq \emptyset$;
- undefined, otherwise.

We have 2^{n-1} unreachable states since for some $q_1 \in Q$ and $w_1 \in L_2$, $\delta(q_1, w_1)$ **must** be defined. □

Proof for Upper Bound

Incomplete DFAs

Proof.

The transitions of γ are defined by setting, for $(r, R) \in P$ and $b \in \Sigma$, we define $\gamma((r, R), b)$ to be

- $(\delta'(r, b),\ \ \delta(R, b) \cup \delta(\delta(r, b), L_2))$, if $r \in Q$ and $(\delta'(r, b) \neq \text{dead}$ or $\delta(R, b) \cup \delta(\delta(r, b), L_2) \neq \emptyset)$;
- $(\text{dead}, \delta(R, b))$, if $r = \text{dead}$ and $\delta(R, b) \neq \emptyset$;
- undefined, otherwise.

We have 2^{n-1} unreachable states since for some $q_1 \in Q$ and $w_1 \in L_2$, $\delta(q_1, w_1)$ **must** be defined. □

Necessary Condition for Upper Bound

Corollary

Let $A = (Q, \Sigma, \delta, q_0, F)$ *be an incomplete DFA with* n *states,* $L_1 = L(A)$ *and* L_2 *is an arbitrary language. Then a necessary condition for* $\mathrm{isc}(L_1 \rightsquigarrow L_2)$ *to reach the upper bound* $(n+1) \cdot 2^n - (2^{n-1} + 2)$ *is that*

$$(\exists q \in Q) \ [\, |\delta(q, L_2)| = 1 \textit{ and } (\forall p \in Q, p \neq q) \ \delta(p, L_2) = \emptyset \,].$$

Note

The conditions of the above corollary do not force L_2 to be a singleton set, **however**, we can achieve simper lower bound proof using a singleton set as L_2.

Lower Bound for Complete DFAs

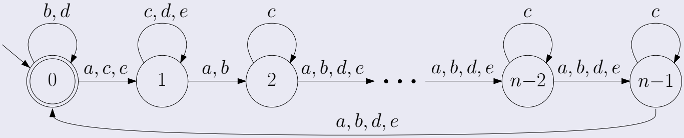
Lemma

Let $\Sigma = \{a, b, c, d, e\}$. *For every* $n \geq 3$ *there exists a complete DFA* A *over* Σ *with* n *states such that*

$$\mathrm{sc}(L(A) \rightsquigarrow \{c\}) = n \cdot 2^{n-1}.$$

Proof.

A complete DFA A



□

Tightness of the Upper Bound

Theorem

For languages $L_1, L_2 \subseteq \Sigma^*$ *where* L_1 *is regular,*

$$\mathrm{sc}(L_1 \rightsquigarrow L_2) \leq \mathrm{sc}(L_1) \cdot 2^{\mathrm{sc}(L_1)-1}.$$

For every $n \geq 3$ *there exists a regular language* L_1 *over a five-letter alphabet with* $\mathrm{sc}(L_1) = n$ *and a singleton language* L_2 *such that in the above inequality we have equality.*

Lower Bound for Incomplete DFAs

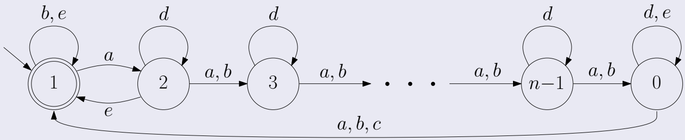
Lemma

Let $\Sigma = \{a, b, c, d, e\}$*. For every* $n \geq 4$ *there exists a regular language* $L_1 \subseteq \Sigma^*$ *recognized by an incomplete DFA with* n *states such that*

$$\mathrm{isc}(L_1 \rightsquigarrow \{c\}) = (n+1) \cdot 2^n - (2^{n-1} + 2).$$

Proof.

An incomplete DFA A



□

Tightness of the Upper Bound

Theorem

For languages $L_1, L_2 \subseteq \Sigma^*$ where L_1 is regular,

$$\text{isc}(L_1 \rightsquigarrow L_2) \leq (\text{isc}(L_1) + 1) \cdot 2^{\text{isc}(L_1)} - (2^{\text{isc}(L_1)-1} + 2).$$

For every $n \geq 4$ there exists a language L_1 over a five-letter alphabet recognized by an incomplete DFA with n states and a singleton language L_2 such that in the above inequality we have an equality.

Conclusions

Results

- We have established tight state complexity bounds for the deletion operation.
- Both in the case where the languages are represented by
  - complete DFAs and
  - incomplete DFAs.
- The state complexity bounds are
  - $n \cdot 2^{n-1}$ — For complete DFAs
  - $(n+1) \cdot 2^n - (2^{n-1} + 2)$ — For incomplete DFAs
- For the lower bound, we used a **five-letter** alphabet.

Can be reduced!

Future Works

Bipolar deletion

The set of trajectories $d^*i^*d^*$ defines the **bipolar deletion** operation. Then, the language $L_1 \rightsquigarrow_{d^*i^*d^*} L_2$ consists of the strings v such that for some string $u = u_1u_2 \in L_2$, the string $u_1vu_2 \in L_1$.

- It is known that bipolar deletion preserves regularity but the current state complexity bound is not optimal.

  2^{3mn} upper bound (M. Domaratzki, 2004)

- Tight bound for the state complexity of bipolar deletion?


תודה
Dankie
Gracias
Спасибо
شكراً
Merci
Takk
Köszönjük
Terima kasih
Grazie
Dziękujemy
Dėkojame
Ďakujeme
Vielen Dank
Paldies
Kiitos
Täname teid
谢谢
Thank You
Tak
感謝您
Obrigado
Teşekkür Ederiz
감사합니다
Σας Ευχαριστούμ
ขอบคุณ
Bedankt
Děkujeme vám
ありがとうございます
Tack